Combining Discrete Wavelet and Cosine Transforms for Efficient Sentence Embedding

Rana Salama[1,3], Abdou Youssef[1], and Mona Diab[2]

[1] School of Engineering and Applied Science, George Washington University
[2] Language Technologies Institute, Carnegie Mellon University
[3] Faculty of Computers and Artificial Intelligence, Cairo University

Abstract. Wavelets have emerged as a cutting edge technology in a number of fields. Concrete results of their application in Image and Signal processing suggest that wavelets can be effectively applied to Natural Language Processing (NLP) tasks that capture a variety of linguistic properties. In this paper, we leverage the power of applying Discrete Wavelet Transforms (DWT) to word and sentence embeddings. We first evaluate, intrinsically and extrinsically, how wavelets can effectively be used to consolidate important information in a word vector while reducing its dimensionality. We further combine DWT with Discrete Cosine Transform (DCT) to propose a non-parameterized model that compresses a sentence with a dense amount of information in a fixed size vector based on locally varying word features. We show the efficacy of the proposed paradigm on downstream applications models yielding comparable and even superior (in some tasks) results to original embeddings.

Keywords: Word Embedding, Sentence Embedding, Discrete Wavelets Transform, Discrete Cosine Transform

1 Introduction

Word embeddings are the basic building blocks for all recent NLP systems. They represent words in distributed dense real-valued vectors which geometrically encode the semantic and syntactic information of words in addition to their linguistic regularities [26]. It has been noted that word embeddings have interesting algebraic characteristics capturing analogies and word relationships suggesting that dimensions along which words are aligned are correlated [29]. These embeddings have a large mean vector and most of their salient features are located in a subspace of much fewer dimensions [35]. This representation has led many to view embeddings as a signal on which spectral techniques are applicable [23]. Such spectral techniques transform data into a new space that captures their different characteristics and sketch new potentials for their usage and the information they convey. Recent spectral analysis in NLP include Discrete Cosine Transform (DCT) [2] and Higher-order Dynamic Mode Decomposition (HODMD) EigenSent [23] embedding models. In this setting, a sentence is represented as a signal with transitional properties captured in the frequency domain using uncorrelated coefficients to encode a sentence.

Such models proved to capture structural variation without losing on efficiency (comparable to averaging) [45] and outperform more complex sentence embedding models [28]. However, these models tend to analyze embedding features along similar word embedding dimensions, on a vertical level (inter-word embedding, aka across all words), accumulating a limited number of base frequency coefficients and dropping the rest, in addition to ignoring their position in the original domain, i.e., ignoring intra-word spectral frequencies within individual word embeddings.

Accordingly, in this paper we propose applying a successful method from image and signal processing, namely, Discrete Wavelet Transform (DWT), that proved to be very effective, fast, and space-efficient in Image Processing [7], Signal Processing [38, 36] and Speech Recognition [39].

We primarily shed light on how DWT can be beneficial for NLP applications. We further adapt DWT to analyze data in terms of their spectral frequency with respect to their position in embedding representations to understand how data vary across different word embedding dimensions. Additionally, we recognize features that are highly varying and those that are nearly identical and can be combined into fewer dimensions.

We posit DWT as an efficient method for word and sentence encoding. We explore DWT to selectively compress relevant information in a word embedding and effectively pack sentence embedding models with semantically condensed word representations. By applying DWT, we avert some of the observed drawbacks of previous spectral models that generate sentence embedding with exponentially increased dimensionality.

Our key contributions are:
1) We propose a novel approach for leveraging DWT to word embeddings that selectively and efficiently compresses relevant salient information at different levels of detail;
2) We show the efficacy of our proposed adapted DWT approach on textual similarity as well as various downstream tasks using non-parameterized word embeddings;
3) We propose the novel approach of conjointly modeling DWT with DCT to encode variable length sentences into a fixed-size vector without hurting performance.

1.1   Motivation

Wavelets Transforms (WTs) are mathematical transformation functions that switch between different levels of detail, or different resolution, to capture various insights about the data. As described in [17], WTs are like a "window", investigating data, with a large "window" would notice coarse features, while looking at the data with a small "window" would notice small features. Accordingly, WT analysis captures
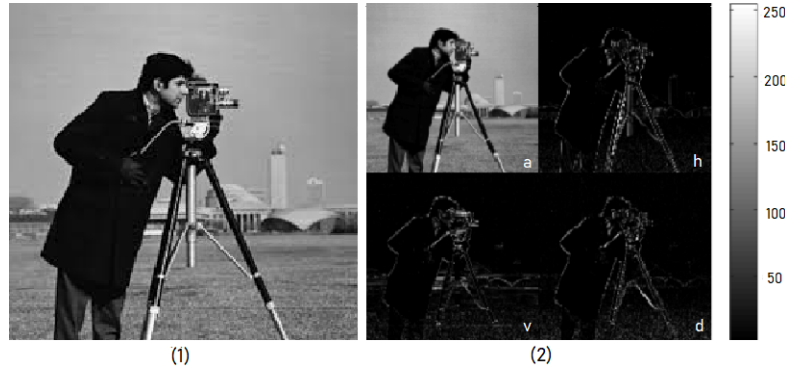
Fig. 1. An example of Level-1 Wavelet transform using DWT to the cameraman image (1) and the corresponding transformed image (2) The transform, (2), produces 4 sub-bands. The top left sub-band is a low resolution version of the original sub-sampled in both horizontal and vertical directions, while the horizontal, h, vertical, v, and diagonal, d, details represent the down-sampled residual versions of the original image.

both the forest and the trees. In Image Processing, as shown in Figure 1, WT transforms the input iconic cameraman picture to four sub images: the top left sub image (or sub-band) captures a low resolution form of the main image (Approximation), comprising low-frequency content, and three other sub-bands comprising high frequency content (Details), that represent different edges present in the image along different directions, horizontal, vertical and diagonal. Accordingly, WTs have been effectively used for image compression [24, 18], and high-frequency components have been used for edge detection [43, 44]. In signal processing, WTs have been used for signal compression [10, 12] [38, 36], speech recognition [39] and noise filtering [14].

From Image and Signal Processing to NLP. WTs can provide considerable insight to NLP as it had in Image and Signal Processing. WTs are expected to provide more insight about the data which should be of great value to NLP. WT can be used to analyze words and sentence representations based on spectral frequency of their variation across time, i.e. where they appear in the vector. Basically, WT can be used to build models that generate arbitrarily good approximations of word representations eliminating irrelevant details, hence generating compressed representations. Also, WT can be used to derive high-frequency representations, which capture details, nuances and contrasts between different features, resulting in equally minimized representations suitable for applications where contrast and subtle nuances are of special value.

1.2   DCT Sentence Embedding

Sentence Embedding is considered important for transferring knowledge to downstream tasks in NLP [4]. Recent advances in sentence embedding proposed using DCT coefficients to compress word vectors considering order and structure of words in a sentence [2]. Later studies proved the efficiency of DCT embedding to capture semantic regularities [45] and demonstrated how it outperforms more complex sentence embedding models [28].

However, DCT sentence embedding is mainly based on the idea of concatenating the first K DCT coefficients, resulting in a sentence vector of size Kd, where d is the word vector dimension, which constitutes a very long representation as the number of coefficients increases. For example, the proposed model in [2] achieved its best performance over a number of tasks using K=6, which results in a sentence vector of size 1800 when using a word vector size of 300. This increased sentence embedding dimension and its impact on the performance of NLP tasks has been a recent research question [42] suggesting that the increased size of a sentence embedding is usually sub-optimal.
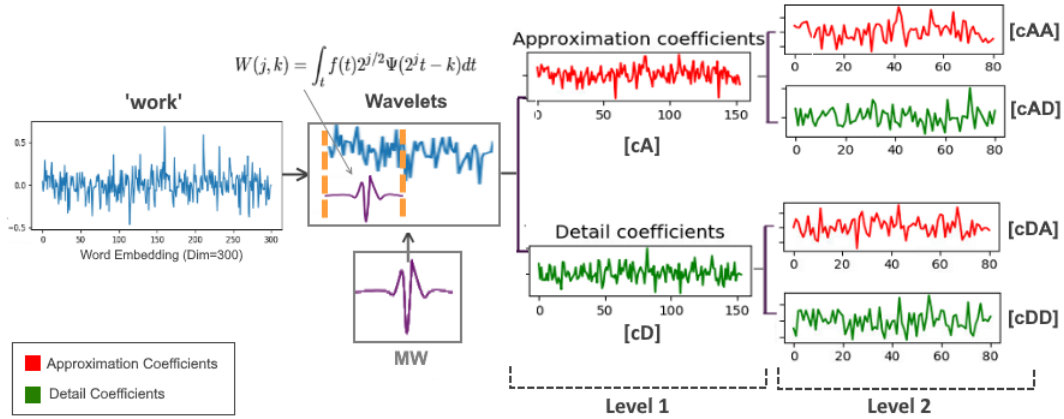

Fig. 2. Plotting word embedding of the word 'work' and its coefficients across 2 levels of WT. cD and cA represent Detail and Approximation coefficients derived from Level-1 of the transform. cDA and cDD indicate Level-2 Approximation and Detail coefficients derived from applying DWT on the Level-1 Detail coefficients. cAA and cAD indicate Level-2 Approximation and Detail coefficients derived from applying DWT on the Level-1 Approximation coefficients.

2   Wavelet Transforms

WT presents a time-frequency analysis tool that can effectively transform a given function $f(t)$ into another domain making certain features more amenable to study

and analyze[27]. In our context, *f(t)* represents the word embedding vector. A WT is applied by translating and shifting a convolution-like function called the Mother Wavelet (MW) Ψ(t) over *f(t)*. MWs are like a microscope zooming in to see details or zooming out to ignore details and see an approximation of the data.

Basically, WT calculates the correlation between the MW and the word vector at different segments of position. Higher correlation indicates more similarity. Using these correlation values, WT efficiently generates pairs of low-pass (low-frequency/high-correlation) and high-pass (high-frequency/low-correlation) filters. The low-pass filters capture Approximation coefficients(cA), and the high-pass filters capture Detail coefficients(cD). The output of a filter pair is usually down-sampled by two. This filtering+downsampling process can be applied on multiple levels recursively as shown in Figure 2. First level transformation will produce cA and cD coefficients downsampled by 2. The second level transformation will further transform cA into new approximation and details coefficients, denoted cAA and cAD, that are further downsampled by 2 in size. Similarly, cD will be transformed into cDA and cDD.

It should be noted that although WT can be used as continuous and discrete transforms, in the context of this paper we only consider Discrete Wavelets Transforms, DWT, which is more suitable for NLP data, which is discrete, that is, word embeddings are finite-dimensional vectors (of real numbers) rather than continuous (i.e., analog) signals.

Formally, given a MW Ψ(t) that can be scaled by factor j and shifted by k, we get the conjugate of Ψ(t) according to the following equation:

$$\Psi_{j,k}(t) = 2^{j/2}\Psi(2^j t - k)$$

The general DWT, denoted by W(j,k), is then given by

$$W(j,k) = \int_t f(t) 2^{j/2}\Psi(2^j t - k)dt$$

To convert data back from the wavelet domain to the original domain, an inverse DWT can be applied.

Note that those definitions are continuous convolutions. In the discrete case as in NLP, discrete convolution counterparts are used. Therefore, WT can be compared to Convolutional Neural Networks (CNNs) [19] in that they both use sliding-window filters and downsampling (pooling). The difference is that in CNNs, the filters are learned from the training data, whereas in WT the filters are designed, not learned. Though less tuned to the data, WT are more computationally efficient in terms of time and space. In fact, some WT algorithms, such as Mallat's pyramid, are linear

in time and space [25].

There are many families of WT functions available in the literature and used as MWs including: Haar, Symmlets, Coiflets, Daubechies, and Biorthogonal, to name a few [27]. Yet as a proof of concept in this paper, and in the interest of space efficiency, we will only be using a subset of the MWs in our experiments. [4]

Comparing DWT to other transforms such as Discrete Cosine Transforms (DCT), DWT allows good localization in both time and frequency domains permitting spectrum analysis of the data, in addition to its spectral behavior in time. DWT allows analyzing related features in word embeddings by their frequency with respect to where they occur. In DCT, feature frequencies are analyzed based on their variation regardless of their location in a word embedding. As a result, DWT can give more analytical insights about the data being transformed. DWT also proved to yield higher compression ratios at comparable fidelity, as well as inherent scaling [36].

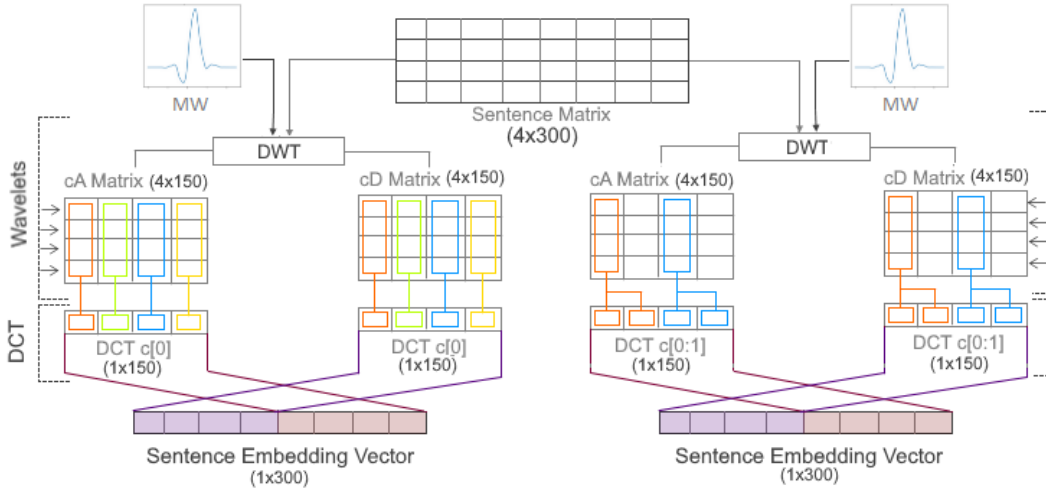**Fig. 3.** Proposed Wavelet-DCT sentence embedding model applied to a sentence of 4 words represented as a matrix. Level-1 DWT and DCT-1 are used as demonstrated on the left side, while the right side demonstrates using Level-1 DWT with DCT-2 by skipping every-other coefficient in DWT coefficients matrix to generate sentence embedding representation.

---

[4] For a more detailed explanation of WT theory, refer to [13] [27] [9].

3   Wavelet-DCT Sentence Embedding

We propose a novel method of applying wavelets conjointly with DCT for sentence embedding. WT can be used to reduce the dimensionality of a word vector retaining important features with respect to local variation, while DCT coefficients are further considered to preserve word order in a sentence.

Given a sentence of N d-dimensional word vectors, $w_1, .., w_N$, we initially stack the words in an $N \times d$ matrix. The sentence matrix is then DWT-transformed row-wise, where every row represents a word, for L levels (L can be 1, 2, ..., n) to compress word embeddings. This transformation will break the sentence matrix into 2^L matrices each of size $N \times \frac{d}{2^L}$, representing the WT coefficients. For L=1, we will have two $N \times \frac{d}{2}$ matrices, one for cA and one for cD. For L=2, we will have four matrices of size $N \times \frac{d}{4}$, namely, cDD, cDA, cAD and cAA. Since we want the sentence embedding to further inherit the details and coefficients of its words, we will keep all DWT-coefficients, but we will further summarize their pattern along different words using a DCT. Applying DCT to summarize DWT coefficients is like the crème de la crème for the N words in a sentence.

Accordingly, the DWT-coefficients matrices will be transformed, column-wise, using DCT. We finally encode a sentence vector of size d by concatenating either 1 or 2 DCT coefficients, which proved to be sufficient empirically and also consistent with observations in [2], for final sentence representation. As mentioned in [2] we will be using K to denote the number of coefficients and $c[0 : K - 1]$ for their actual values. When the number of DCT coefficients is 1; $K = 1$, the DCT-DWT-coefficients matrices will be of size $1 \times \frac{d}{2^L}$ each, concatenating them results in a final sentence embedding of size d.

However, when $K = 2$, we generate our final sentence embedding vector by using a simple *trick* that skips every other DWT coefficient before applying DCT. This will further reduce a DWT-coefficient matrix into $N \times \frac{d}{2^{L+1}}$. As a result, selecting 2 DCT coefficients and concatenating them will still result in a sentence vector of size d. This trick proved, empirically, not to affect the information preserved by these coefficients but keep the sentence embedding dimension unchanged.

4   Evaluation of DWT Efficacy

We initially investigate the efficacy of applying DWT to word embeddings. We explore DWT coefficients and selectively using them as new compressed embeddings. We evaluate the efficacy of the new embeddings to capture and compress semantics using semantic similarity tasks. We further evaluate their effectiveness in a number of downstream tasks to evaluate their efficacy extrinsically.

In this evaluation, we will consider the following DWT coefficients as embeddings; Level-1 Detail (cD) and Approximation (cA) coefficients. Additionally, we extend these coefficients with Level-2 coefficients (performed on Level-1 Approximation and Detail coefficients as shown in Figure 2), $cD + cAD$ and $cA + cDA$, to enrich the word representation with more information from subsequent transformation levels.

4.1   Experimental Setup

Embeddings For our experiments we used a number of base embeddings to demonstrate the capabilities of DWT. In this context we use GloVe [34] and Fast-Text[28] for word embeddings with various dimensions (100, 200, 300) for a comprehensive evaluation.

Sentence Embedding Evaluation For sentence embeddings, we use the SentEval toolkit [11] for evaluation. For all downstream tasks, we employed multi-layer perceptron (MLP) classifiers based on the setup outlined in SentEval.

4.2   Intrinsic Evaluation

We evaluate DWT embeddings and their encoded semantics in two evaluation tasks: Word Similarity and Concept Categorization. For Word Similarity, we use the following datasets : SimLex-999 [20], MEN[8] and WS353[16].

Baseline We set the original embeddings, i.e. without transformation, as our baseline: F-BASELINE and G-BASELINE1, corresponding to FastText and GloVE embeddings. In GloVE, we consider another baseline, G-BASELINE2, which is the original GloVE embedding with 50% less dimensions to compare the reduced DWT embeddings to the original embeddings of the same size, i.e. for a GloVE embeddings of size 100, the transformed DWT embeddings will be of size 50, so we consider GloVE embedding of size 50 as another baseline.

Our experiments show that Level-1 DWT embeddings, cA and cD, with 50% reduction in size, surpass the performance of the baselines, G-BASELINE1 and G-BASELINE2, as depicted in Table 1. Given the original dimension of 100, for the SimLex dataset, cA outperforms G-BASELINE1 by 8%. Moreover, cA exhibits superior performance compared to G-BASELINE2 and increases performance by 11% even though they share the same size. This empirically demonstrates that DWT embeddings capture the underlying semantics conveyed in the embeddings and effectively encoding them into fewer dimensions. While for WS353 and MEN

datasets, cD beats both baselines, illustrating that for different datasets, every set of coefficients captures different aspects about the data.

Furthermore, for dimension of 200, cA outperforms specifically for the SimLex and MEN datasets, emphasizing how various coefficients capture information within the embeddings as the dimension size increases.

In our subsequent experiment, we utilize FastText embeddings with 300 dimensions as the baseline, F-BASELINE. Furthermore, we explore Level-2 DWT embeddings, specifically, we enrich the Level-1 cA coefficients with the approximation coefficients from the Level-2 cD transform, cDA. Similarly, we examine the Level-1 cD coefficients alongside the Level-2 detail coefficients for Level-1 approximations, cAD. As shown in Table 2, at a compression of 25% dimensionality reduction, cD+cAD yields comparable results to F-BASELINE for all 3 datasets. For the SimLex dataset, the cD coefficients effectively encode the semantics, and the addition of coefficients from subsequent layers (cD+cAD) didn't improve the performance. [5] However, for the MEN and WS353 datasets, combining the coefficients (cD+cAD) achieves performance comparable to the F-BASELINE embeddings, with a slight reduction of at most 1% in performance.

Table 1. Word Similarity Evaluation using GloVe Embeddings

	Dim	SimLex	WS353	MEN
G-BASELINE1	**100**	0.12	0.46	0.57
G-BASELINE2	**50**	0.09	0.42	0.53
cD	50	0.11	**0.50**	**0.58**
cA	50	**0.2**	0.44	0.57
G-BASELINE1	**200**	0.13	**0.48**	0.59
G-BASELINE2	**100**	0.12	0.46	0.57
cD	100	0.13	0.40	0.48
cA	100	**0.17**	0.47	**0.62**

Spearman Rank Order Correlation (SPC) Results on SimLex-999, WS353 and MEN datasets; using GloVe-Twitter27B embeddings with dimensions 100 and 200 compared to Level-1 DWT coefficients on word similarity tasks. In addition to G-BASELINE2 which represents GloVE embedding with 50% less dimensions. cD and cA correspond to the embeddings yielded at Level-1 DWT transform. Best results are in bold and best results per experimental condition are in red.

We further evaluate DWT embeddings using the Concept Categorization task that groups words in different categories based on semantic clusters [5]. Such models have been proven effective in downstream NLP tasks [41]. We evaluate DWT embeddings using the AP [3], BM [31] and BLESS datasets [6]. We use FastText as original embeddings. As illustrated in Table 3, cA and cD embeddings yield comparable and

---

[5] In this context, cDD will represent the details of the details of Level-1 and hence won't add much information

Table 2. Word Similarity Evaluation using FastText Embeddings

	Dim	SimLex	WS353	MEN
F-BASELINE	**300**	**0.5**	**0.79**	**0.83**
cD	150	0.5	0.73	0.79
cA	150	0.49	0.74	0.79
cD+cAD	225	0.5	0.78	0.82
cA+cDA	225	0.49	0.75	0.82

even superior results compared to F-BASELINE with a dimensionality reduction of 50%.

Table 3. Concept Categorization Evaluation

	Dim	AP	BM	BLESS
BASELINE	300	**0.70**	0.47	0.86
cD	150	0.70	0.46	0.87
cA	150	0.70	0.49	0.82

Furthermore, we study DWT embeddings qualitatively, we take a snippet of different word pairs and their cosine similarity measures using FastText embeddings, cA and cD embeddings as shown in Table 4. The results show that, on average, cD performs as well as original word embeddings. However, for cA, some interesting results hold: For pairs like, 'dog-cow' and 'happy-cry', cA embeddings tend to outperform with a big margin both the original embeddings and the cD coefficients. This could typically indicate that these two words have a high correlation between their Approximation coefficients and basically capture a different type of relation, for example, 'dog' and 'cow' are both animals, and 'happy' and 'cry' are both emotions. It could also indicate that the low-pass filtering (which produces the cA's) preserves and highlights the commonality between 'dog' and 'cow' (their animal nature), and between 'happy' and 'cry' (their emotional nature), leading to higher similarity, whereas the high-pass filtering (which produces the cD's) eliminates/reduces those kinds of commonalities, but preserves certain other connections that yield the same (or even slightly better) similarity value as/than the original embeddings, but lower similarities than those produced by cA.

On the other hand, for the 'boy-girl' and 'woman-girl' pairs, cA decreases the similarities drastically, while cD does not, indicating that the subtle nuanced connection/contrast (of gender or age) got lost in the low-pass filtering but was preserved by the high-pass filtering. These curious observations call for further investigation into the working of the low-pass filtering and high-pass filtering of WT, a subject for future research.

Table 4. Similarity measures for different word pairs

Word Pairs	Word Vector	cD	cA
'boy'-'girl'	0.77	0.78	0.57
'dog'-'cow'	0.39	0.39	0.89
'woman'-'girl'	0.65	0.67	0.49
'happy'-'cry'	0.28	0.37	0.91

We additionally captured the 5-nearest neighbors for a set of randomly selected words, as shown in Table 5. cD and cA were able to capture some linguistic phenomena that the original word embeddings missed. For example, 'Cat' was not one of the 5-nearest neighbors for 'cat'. Synonyms like 'contented' are closer to happy as opposed to 'unhappy' in the original representation. When using cA, we capture more synonyms of 'happy' like 'thrilled' and 'overjoyed', and when using cD, we capture arguably better relational similarity to 'Italy' such as 'sicily' and 'levorno', which are Italian cities.

Table 5. 5-nearest cosine similar words

Word	Word Vector	cD	cA
'cat'	'cats', 'kitty', 'kitten', 'feline', 'kitties'	cats', 'kitten', 'kitty', 'feline', 'Cat'	'cats', 'kitty', 'kitten', 'feline', 'kitties'
'happy'	'happpy', 'unhappy', 'hapy', 'contented', 'happier'	'happpy', 'contented', 'Happy', 'hapy', 'unhappy'	'thrilled', 'happpy', 'unhappy', 'overjoyed', 'hapy'
'Italy'	'rome', 'italy.', 'spain', 'france', 'europe'	'spain', 'sicily', 'italy', 'livorno', 'europe'	'rome', 'italy', 'france', 'switzerland', 'germany'

4.3 Extrinsic Evaluation

For extrinsic evaluation, DWT embeddings are applied in the same settings of intrinsic evaluation to the following downstream classifications tasks through SentEval toolkit [11]: sentiment classification on Movie Reviews (MR), Stanford Sentiment Treebank (SST2, SST5) [32], product review (CR) [21], subjectivity classification (SUBJ) [33], opinion polarity classification (MPQA), question type classification (TREC) [40], paraphrase identification (MRPC) [15], STS12 semantic similarity task [1], and entailment classification on the SICK dataset (SICK-E) [7]. We use DWT embeddings to encode sentences then feed the encoded sentences to the aforementioned tasks.

Baseline In this evaluation, we consider averaging as the baseline sentence encoding (AVG), using FastText word embeddings. We further consider other baselines to demonstrate the efficacy of DWT embeddings including: (1) FastText embeddings

with random 50% dimensionality reduction, denoted as random pooling, under the assumption of no correlation between dimensions. This involves the random elimination of 50% of the features in a word embedding.(2) An embedding model that is based on a dimensionality reduction method, namely PCA(Principal component analysis), as developed in [35] and denoted as (PCA+PPA). This embedding combines PCA dimensionality reduction embeddings with a post-processing algorithm (PPA) [30] to construct effective word embeddings of lower dimensions.

Table 6. Extrinsic Results

Embedding	Dim	Sentiment Analysis					Inference	Paraphrase		SUBJ	TREC
		MR	CR	SST2	SST5	MPQA	SICK-E	MRPC	STS12		
AVG	300	78.3	79.6	**84.13**	44.16	87.94	**79.5**	74.43	58.3	92.33	83.2
AVG(Random Pooling)	150	58.83	63.58	56.07	28.82	69.61	56.59	67.07	8	69.78	33
AVG(PCA+PPA)	150	75.52	78.2	-	41.4	86.18	75.06	73.39	53.34	90.96	75.2
AVG(PCA+PPA)	200	77.18	79.76	-	43.48	86.64	76.76	72.93	53.76	91.6	77.4
AVG(cA)	150	**78.57**	80.85	80.12	43.6	85.61	78.41	73.22	58.01	90.01	82
AVG(cD)	150	$76.61_{coif}15$	79.26	80.62	43.53	86.41	78.46	73.62	57.71	91.97	78.6
AVG(cD+cAD)	225	78	80.69	80.94	44.7	87.89	78.53	73.86	**58.34**	92.12	81.8
AVG(cA+cDA)	225	77.4	**81.22**	82.43	**45.52**	87.36	78.49	73.49	57.94	91.96	83.2
AVG(cD+cAD+cAAD)	263	78.38	81.09	81.55	43.94	**87.95**	78.89	**74.5**	**58.34**	**92.34**	**84.8**

Best Classification accuracy (except STS12, the metric is Pearson correlation %) results on various classification tasks. The Baseline AVG is the simple original word embedding averaging. AVG(PCA+PPA) is the average of a PCA based embedding proposed in   cite82 with dimensions 150 and 200   cite67. AVG(cD) represents the average of Level-1 Detail coefficients, AVG(cD+cAD) is the average of Level-1 Detail coefficients concatenated to Level-2 Detail coefficients as derived from Level-1 Approximation coefficients. Similarly for AVG(cD+cAD+cAAD) and AVG(cD+cAD+cAAD). We also illustrate the MW used per condition per task in italics. The best overall results are shown in bold. Best results per condition are shown in red.

In Table 6, the overall results are comparable to the baselines AVG for most tasks with significant reduction in dimensionality, yet DWT outperforms in CR, SST5, MPQA, MRPC, STS12, SUB and TREC. In this experiment we further consider Level-3 DWT embeddings as shown in the table. For SUBJ, Level-3 yields the best results with a compression rate of 12.5%. For the sentiment classification tasks we note performance comparable to AVG for MR and MPQA, while outperforming AVG for CR and SST5 at Level-2 cA+cDA with compression rate of 25%. We also note that in the SST2 task, none of the conditions beat AVG. Similarly, for the entailment task of SICK-E none of the conditions surpassed AVG. However, for the Paraphrase tasks of MRPC and STS12 as well as the question classification task TREC, Level-3 representations yielded comparable performance to AVG with a compression rate of 12%.

In general, Level-2 AVG(cA+cDA) and Level-3 AVG(cD+cAD+cAAD) yield the best results compared to other conditions except for the MR task where AVG(cA)

yields the highest result (78.57%) followed by the Level-3 condition at an accuracy of 78.38%. Furthermore, if the embedding size is a major concern, observe that with cA or cD alone (at half the embedding size relative to the original), the performance is comparable to the AVG baseline in nearly all the tasks considered. Hence, we conclude that DWT presents an effective balance between efficiency (compactness) and accuracy, and an effective data-size reduction method with hardly any adverse effect on accuracy.

5 Evaluating DWT with DCT Sentence Embedding

We evaluate the effectiveness of DWT by studying the application of DWT to the DCT sentence embedding model. We argue that DWT can encapsulate ample and pertinent information within word embeddings, leading to an improved representation for sentence embeddings without increasing dimension size nor compromising performance.

Baseline In this evaluation, we consider a number of sentence embedding models as a baseline including:

1. Similar spatial models, namely, DCT [2] with 1 and 2 coefficients to be consistent with our proposed model, and EigenSent model [23][6]. EigenSent sentence embeddings are composed by keeping the top m dynamic modes resulting in a sentence embedding of size md ignoring correlation and information encoded within a word embedding for every word in the sentence.
2. Other top performing non-parametrized sentence embedding models: P-Means [37] and VLAWE [22] to comprehend our result analysis.

For a fair comparison, we use multi-layer perceptron (MLP) classifiers based on various settings including number of hidden states (in [0, 50, 100, 200]) and dropout rates (in [0, 0.1]) considering the same settings in [2] that yielded their best results for DCT-based sentence embedding.

5.1 DWT-DCT Experimental Results

The results on different tasks demonstrate the power of applying DWT to word embeddings. The combined DWT-DCT outperforms the baseline AVG in all tasks with a significant margin in some tasks. Comparing our approach to DCT alone, as shown in Table 7, our combined model yields better results. It can be shown that Level-1 or Level-2 DWT with 1 coefficient, DWT1-DCT[0] and DWT2-DCT[0], generally outperforms DCT with 1 coefficient, DCT c[0], in all task except MR which is comparable. Similarly with 2 coefficients, DWT1-DCT[0:1], applying DWT to DCT

---
[6] Embeddings generated using https://github.com/DeepK/hoDMD-experiments

performs better in most of the tasks, except for SUBJ which is comparable, with 50% reduction in size. It is worth mentioning that in CR, our DWT-DCT embeddings outperforms the best result achieved in [2], which is 80.08, when $K = 5$, which results in a final DCT sentence embedding of size 1500. In SICK-E, DCT embedding model achieves a comparable result when $K = 4$, which results in a final DCT sentence embedding of size 1200. Additionally, in SST2, DCT embedding didn't beat the AVG baseline, while our model outperformed it. As for EigenSent, our model significantly surpasses the others, with the same embedding size, across all tasks. Compared to other non-parametrized models, our model surpasses VLAWE in all tasks. As opposed to P-Means, our model outperforms or is comparable to the other models, except for SICK-E and STS16, yet our results are convenient given the fact that our embedding dimensionality is 12 times lower in magnitude.

Our observations suggest that DWT seems to be complementary to DCT as we observe better performance. Additionally, Level-1 DWT is sufficient to pack a sentence embedding with relevant information.

Overall, the combined model is robust and efficient, yielding results that are better than or comparable to the state of the art models across a variety of standard tasks.

Table 7. Experimental results from applying DWT to DCT sentence embeddings

Model	Dim	Sentiment Analysis					Inference	Paraphrase				SUBJ	TREC
		MR	CR	SST2	SST5	MPQA	SICK-E	MRPC	STS12	STS16	STSB		
# Samples	-	11k	4k	70k	19.5k	11k	10k	5.7k	8.1k	3.9k	15k	10k	6k
AVG	300	78.3	79.6	84.13	44.16	87.94	79.5	74.43	58.0	64.0	69.26	92.33	83.2
p-means	3600	78.3	80.8	84	-	**89.1**	83.5	73.2	54	**67**	72	92.6	**88.4**
VLAWE	3000	77.7	79.2	80.8	-	88.1	81.2	72.8	-	-	-	91.7	87
EigenSent	300	70.26	73.16	72.54	36.97	69.15	71.34	70.43	36	-	-	85.73	54.2
DCT c[0]	300	78.45	79.81	83.53	44.57	88.36	78.91	72.93	58.3	64.1	71.5	92.79	84.8
DWT1-DCT[0]	300	78.38	*81.14*	***84.68***	*46.73*	88.36	*79.5*	*73.33*	*58.6*	*64.5*	*71.6*	***92.8***	84.6
DWT2-DCT[0]	300	78.31	***81.27***	*83.86*	*45.93*	88.35	*79.58*	*73.51*	***58.73***	*64.8*	***72.3***	***92.8***	*85*
DCT c[0:1]	600	78.15	79.84	83.47	46.06	87.76	79.64	72.81	50.4	57.4	71.2	92.61	88.2
DWT1-DCT[0:1]	300	***78.57***	*80.66*	*83.91*	***46.74***	87.93	***80.9***	***74.5***	*50.8*	57.4	*71.4*	92.51	***88.4***

Results as opposed to DCT embedding, DCT c*, as reported in [2]. Our model names use the convention 'DWT*l*-DCT[*k*]' to indicate *l*-Level DWT followed by DCT where coefficients *k* of the DCT output are taken for the final representation; e.g., DWT1-DCT[0] is used for Level-1 of DWT and c[0] DCT coefficient. 'Dim' reflects the sentence embedding vector size. In STS12, STS16 and STSB Pearson correlation coefficients(%) is used. The best overall results are shown in bold. Best results per condition are shown in red. Results where the DWT-DCT sentence embedding based model exceeds the DCT model are shown in italic.

6   Conclusion and Discussion

The results on different tasks demonstrate the power of our method. Our combined DWT and DCT outperforms the baseline AVG (in the majority of cases, by a

significant margin) across all tasks. It outperforms p-means and VLAWE as well, despite their several orders of magnitude larger dimensionality for both p-means and VLAWE. Comparing DCT to our combined models, the combined models yielded better results especially compared to DCT[0:1] and DCT[0:2] (except for TREC) despite having larger dimensionality for DCT. We also note that DWT seems to be complementary to DCT as we observed better performance in the combined models except for MR where the best results are the same, and for TREC where DCT[0:2] outperforms all the combined models. Overall, our combined models are robust and efficient, yielding results comparable or even outperforming the state of the art of non-parameterized results across a variety of standard tasks.

6.1   Mother Wavelets

While we simplified our paper by omitting the specific details of MW and its application in each task, it remains a crucial element of WT. In our experiments, we explored the use of Coiflets, Daubechies, and Symlets wavelets at different scales; i.e. how stretched is the MW across the data. We observed that Coiflets, on average, demonstrate strong performance across various tasks and scales of expansion. This suggests that the Coiflets wavelets fits well the structure of the embedding data.

Intuitively, we surmise from our experiments that frequent words will tend to transfer better with wide scaled wavelets to capture approximation details more efficiently. Conversely, less frequent, specialized words in a given context may yield better results when transformed using smaller-scaled wavelets. The selection of the best MW and scale to be used requires more research and we plan to investigate this more in our future work.

6.2   Correlation

Although we did not explicitly prove that the dimensions in the same embedding are correlated, the results of applying DWT suggests that there exists a correlation between these dimensions. If no correlation existed, DWT would never achieve comparable results to original embeddings' performance.

6.3   Computational Complexity

The computational complexity of applying DWT to embedding is typically linear. The general complexity is often expressed in terms of the dimension of the embedding, denoted as N and depends on the number of levels in the DWT. If the transform has L levels, the overall complexity is

$$O(L \times N)$$

Accordingly, using DWT for embeddings analysis and compression is computationally efficient.

6.4   Conclusion

In this paper we explored the effectiveness of applying DWT to word embeddings to *selectively* reduce word embeddings into approximated or detailed coefficients representations by exploring frequency and space analysis of a word embedding, retaining the same performance at 50% to 75% of the word vector original dimension size. The generated DWT word embedding postulates that different sets of coefficients capture different semantic aspects of a word embedding. Our intrinsic and extrinsic evaluations for the efficacy of applying DWT in the context of NLP suggest that DWT has significant potential for *efficiently* modeling word and sentence embeddings.

We further use the resulting word embeddings to generate a DWT-DCT based sentence-embedding. The proposed embedding method not only yields comparable or even better performance than original embedding models, it also has the added advantage of significantly reducing sentence vector size, whittling it down to salient info for the task. Finally, our model is able to outperform vector averaging for the SST Task which to date is the dominating model among non-parametric models.

References

1. Eneko Agirre, Daniel Cer, Mona Diab, and Aitor Gonzalez-Agirre. SemEval-2012 task 6: A pilot on semantic textual similarity. In **SEM 2012: The First Joint Conference on Lexical and Computational Semantics – Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*, pages 385–393, Montréal, Canada, 7-8 June 2012. Association for Computational Linguistics.
2. Nada Almarwani, Hanan Aldarmaki, and Mona Diab. Efficient sentence embedding using discrete cosine transform. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3672–3678, Hong Kong, China, November 2019. Association for Computational Linguistics.
3. Abdulrahman Almuhareb. Attributes in lexical acquisition. 2006.
4. Sanjeev Arora, Yingyu Liang, and Tengyu Ma. A simple but tough-to-beat baseline for sentence embeddings. January 2019. 5th International Conference on Learning Representations, ICLR 2017 ; Conference date: 24-04-2017 Through 26-04-2017.
5. Marco Baroni, Georgiana Dinu, and Germán Kruszewski. Don't count, predict! a systematic comparison of context-counting vs. context-predicting semantic vectors. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 238–247, Baltimore, Maryland, June 2014. Association for Computational Linguistics.
6. Marco Baroni and Alessandro Lenci. How we blessed distributional semantic evaluation. pages 1–10, 07 2011.
7. Toufik Bouden and Mokhtar Nibouche. *The Wavelet Transform for Image Processing Applications*. 04 2012.
8. Elia Bruni, Nam Khanh Tran, and Marco Baroni. Multimodal distributional semantics. *J. Artif. Int. Res.*, 49(1):1–47, January 2014.
9. Steven L. Brunton and J. Nathan Kutz. *Data-Driven Science and Engineering: Machine Learning, Dynamical Systems, and Control*. Cambridge University Press, 2019.

10. Ronald Coifman, Yves Meyer, Steven Quake, and Mladen Wickerhauser. Signal processing and compression with wavelet packets. *Tech rep*, 01 1991.

11. Alexis Conneau and Douwe Kiela. Senteval: An evaluation toolkit for universal sentence representations. *CoRR*, abs/1803.05449, 2018.

12. Julián L Cárdenas-Barrera, Juan V Lorenzo-Ginori, and Ernesto Rodríguez-Valdivia. A wavelet-packets based algorithm for eeg signal compression. *Medical Informatics and the Internet in Medicine*, 29(1):15–27, 2004.

13. Ingrid Daubechies. *Ten Lectures on Wavelets*. Society for Industrial and Applied Mathematics, USA, 1992.

14. Ç. P. Dautov and M. S. Özerdem. Wavelet transform and signal denoising using wavelet method. In *2018 26th Signal Processing and Communications Applications Conference (SIU)*, pages 1–4, 2018.

15. Bill Dolan, Chris Quirk, and Chris Brockett. Unsupervised construction of large paraphrase corpora: Exploiting massively parallel news sources. In *COLING 2004: Proceedings of the 20th International Conference on Computational Linguistics*, pages 350–356, Geneva, Switzerland, aug 23–aug 27 2004. COLING.

16. Lev Finkelstein, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman, and Eytan Ruppin. Placing search in context: The concept revisited. volume 20, pages 406–414, 01 2001.

17. A. Graps. An introduction to wavelets. 1995.

18. S. Grgic, M. Grgic, and B. Zovko-Cihlar. Performance analysis of image compression using wavelets. *IEEE Transactions on Industrial Electronics*, 48(3):682–695, 2001.

19. Jiuxiang Gu, Zhenhua Wang, Jason Kuen, Lianyang Ma, Amir Shahroudy, Bing Shuai, Ting Liu, Xingxing Wang, and Gang Wang. Recent advances in convolutional neural networks. *CoRR*, abs/1512.07108, 2015.

20. Felix Hill, Roi Reichart, and Anna Korhonen. Simlex-999: Evaluating semantic models with (genuine) similarity estimation. *CoRR*, abs/1408.3456, 2014.

21. Minqing Hu and Bing Liu. Mining and summarizing customer reviews. In *KDD '04*, 2004.

22. Radu Tudor Ionescu and Andrei M. Butnaru. Vector of locally-aggregated word embeddings (vlawe): A novel document-level representation, 2019.

23. Subhradeep Kayal and George Tsatsaronis. EigenSent: Spectral sentence embeddings using higher-order dynamic mode decomposition. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4536–4546, Florence, Italy, July 2019. Association for Computational Linguistics.

24. A. S. Lewis and G. Knowles. Image compression using the 2-d wavelet transform. *IEEE Transactions on Image Processing*, 1(2):244–250, 1992.

25. Tao Li, Qi Li, Shenghuo Zhu, and Mitsunori Ogihara. A survey on wavelet applications in data mining. *SIGKDD Explorations*, 4:49–68, 12 2002.

26. Yang Li and Tao Yang. *Word Embedding for Understanding Natural Language: A Survey*, volume 26. 05 2017.

27. G. Madhavan. The illustrated wavelet transform handbook - introductory theory and applications in science, engineering, medicine and finance [book review]. *IEEE Engineering in Medicine and Biology Magazine*, 22(1):92–93, 2003.

28. Tomas Mikolov, Edouard Grave, Piotr Bojanowski, Christian Puhrsch, and Armand Joulin. Advances in pre-training distributed word representations. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan, May 2018. European Language Resources Association (ELRA).

29. Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 746–751, Atlanta, Georgia, June 2013. Association for Computational Linguistics.

30. Jiaqi Mu, Suma Bhat, and Pramod Viswanath. All-but-the-top: Simple and effective postprocessing for word representations. *CoRR*, abs/1702.01417, 2017.

31. Brian Murphy, Partha Talukdar, and Tom Mitchell. Selecting corpus-semantic models for neurolinguistic decoding. In *Proceedings of the First Joint Conference on Lexical and Computational Semantics - Volume 1: Proceedings of the Main Conference and the Shared Task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation*, SemEval '12, page 114–123, USA, 2012. Association for Computational Linguistics.

32. Bo Pang and Lillian Lee. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, pages 271–278, Barcelona, Spain, July 2004.

33. Bo Pang and Lillian Lee. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, pages 271–278, Barcelona, Spain, July 2004.

34. Jeffrey Pennington, Richard Socher, and Christopher Manning. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar, October 2014. Association for Computational Linguistics.

35. Vikas Raunak, Vivek Gupta, and Florian Metze. Effective dimensionality reduction for word embeddings. In *Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019)*, pages 235–243, Florence, Italy, August 2019. Association for Computational Linguistics.

36. O. Rioul and M. Vetterli. Wavelets and signal processing. *IEEE Signal Processing Magazine*, 8(4):14–38, 1991.

37. Andreas Rücklé, Steffen Eger, Maxime Peyrard, and Iryna Gurevych. Concatenated p-mean word embeddings as universal cross-lingual sentence representations. *CoRR*, abs/1803.01400, 2018.

38. Mallat Stéphane. Chapter 10 - compression. In Mallat Stéphane, editor, *A Wavelet Tour of Signal Processing (Third Edition)*, pages 481 – 533. Academic Press, Boston, third edition edition, 2009.

39. Nitin Trivedi, Vikesh Kumar, Saurabh Singh, Sachin Ahuja, and Raman Chadha. Speech recognition by wavelet analysis. *International Journal of Computer Applications*, 15:27–32, 2011.

40. Ellen M. Voorhees and Dawn M. Tice. Building a question answering test collection. In *Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '00, page 200–207, New York, NY, USA, 2000. Association for Computing Machinery.

41. Bin Wang, Angela Wang, Fenxiao Chen, Yuncheng Wang, and C.-C. Jay Kuo. Evaluating word embedding models: methods and experimental results. *APSIPA Transactions on Signal and Information Processing*, 8:e19, 2019.

42. Hongwei Wang, Hongming Zhang, and Dong Yu. On the dimensionality of sentence embeddings. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 10344–10354, Singapore, December 2023. Association for Computational Linguistics.

43. Z. Xizhi. The application of wavelet transform in digital image processing. In *2008 International Conference on MultiMedia and Information Technology*, pages 326–329, 2008.

44. Zhen Zhang, Siliang Ma, Hui Liu, and Yuexin Gong. An edge detection approach based on directional wavelet transform. *Computers Mathematics with Applications*, 57(8):1265 – 1271, 2009.

45. Xunjie Zhu and Gerard de Melo. Sentence analogies: Exploring linguistic relationships and regularities in sentence embeddings, 2020.